في هذه المحاضره ستكون متعة التعامل مع المايكروكنترولر .... حيث سنتحدث عن الاتصال بين المايكروكنترولر وجهاز الكمبيوتر .... وتناقل الداتا خلالهما ....

ان عملية الاتصال هذه تسمى الاتصال المتسلسل Serial Communication لكن ماذا يعني اتصال متسلسل؟ لنعرف الاتصال المتسلسل بطريقه مبسطه .

الاتصال المتسلسل : هو بروتوكول يستخدم لتوصيل بين جهازين أو أكثر ... ويمكن أن يكون الجهاز هو الكمبيوتر

لكن ماذا نعني بـ بروتوكول ؟
البروتوكول : هو مجموعة من القواعد التي تحكم موضوع معين .... موضوع مثل عملية الاتصال التي سنتحدث عنها في هذه المحاضرة .

اذا بلغة اخرى من الممكن تعريف الاتصال المتسلسل بأنها طريقة لها مجموعة من القواعد تستخدم لتوصيل بين جهازين أو أكثير وغالباً يكون احد هذه الاجهزه هو الكمبيوتر .

وهنا سيكون الجهازين هما المايكروكنترولر والكمبيوتر ..... اذا الان توصل لنا فكره مهمه بأن الاتصال المتسلسل له مجموعه من القواعد .... اذا ما هي هذه القواعد ؟؟

اهم هذه القواعد سنذكر قاعدتين ..... وهي الـ Logic level وفي الواقع تعتبر الـ Logic level هي احد المشاكل التي تواجه عملية الاتصال بين المايكروكنترولر والكمبيوتر ...ذكرنا سابقاً ان المايكروكنترولر يتعامل فقط مع صفر فولت و ٥ فولت .... ولكن لتوصيل الكمبيوتر نستخد serial port له Logic level مختلف سنعرفها لاحقاً ....

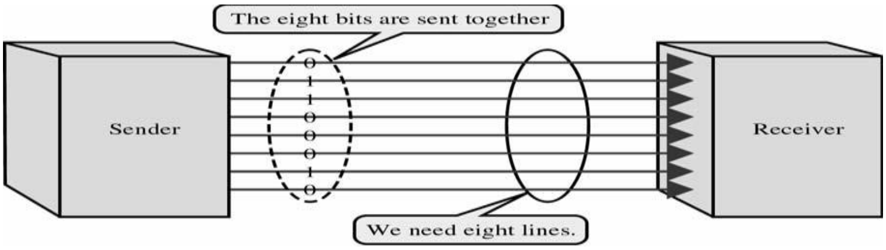
ليست هنا فقط المشكله .... بل فأن كل من الكمبيوتر والمايكروكنترولر لن يستطيع فهم الداتا المرسله له لانها بـ Logic level مختلف تماماً

والمشكله الاخرى التي تواجهنا هي ان المايكروبروسيسر الموجود في المايكروكنترولر يرسل البيانات بشكل Parallel لكن الـ serial port من اسمه فهو سيريال .

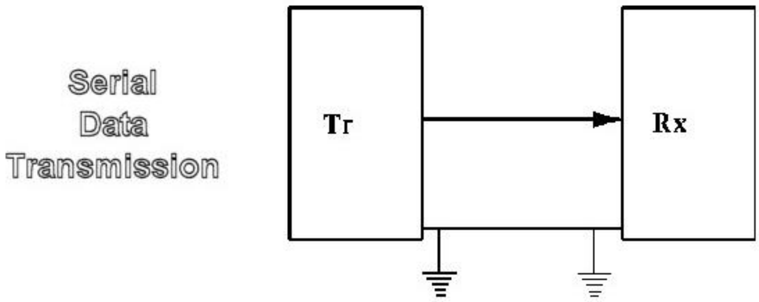
اذا ما الحل .....؟

الحل هو بناء دئره هاردوير ... لكن هذه الدائرة غير موجوده داخل المايكروكنترولر .... فيتطلب منا ان نصممها بنفسنا لتؤدي هذا الوظيفة وتخلصنا من المشاكل التي ذكرت سابقة ... وأهما تحويل الـ logic level الخاص بالمايكروكنترولر الى الـ Logic level الخاص بالـ serial Port .... وهذا سنتعرف عليه بوقته .

لكن دعنا نتعرف الان على الفرق بين ارسال البيانات Parallel وارسالها serial لاحظ الصورة التاليه التي تبين نقل البيانات Parallel حيث ستنتقل البيانات بوقت واحد ... وهذا يسغرق وقت قليل ويتطلب اسلاك اكثر



اما نقل البيانات بشكل serial فهو يحتاج لعدد اقل من الاسلاك ... ولكن يستغرق وقت أطول وهذا موضح في الصورة التالية



تنويه : رغم انه الاتصال الـ serial يستغرق وقت أطول ... لكنه مستخدم بكثره ... والسبب في ذلك ان المايكروكنترولر يتعامل بالمايكروثانية ... فعندما نقول وقت طويل نقصد بذلك طويل بالنسبه لنقل الـ Parallel لكن لا يشاهد الفرق بالنسبه للمستخدم .... لانه وبالطبع لايمكن الشعور بوقت قصير جداً

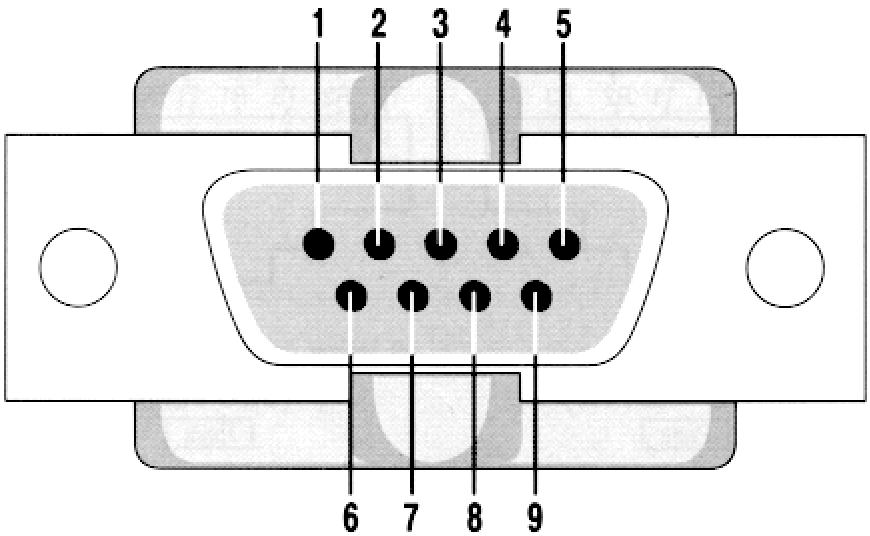
لكن مهلاً ... قد يتسائل البعض اننا ذكرنا كلمة الـ serial port كثيراً ... لكن ما هو ؟؟

الـ serial port هو السلك المستخدم لنقل الداتا وبتأكيد الكثير يعرفه كما في الصورة ... فهو الوسيلة وهو خط الاتصال بين المايكروكنترولر والكمبيوتر .... وبتأكيد باستخدام الدائرة التي ذكرناها سابقاً ... هل تتذكرونها؟



نحتاج الان الى التعرف الـ Pin الموجوده في هذا السلك ... لاننا سنستخدما لتوصيل والصورة التالية تأخذ المقطع الامامي من الـ serial port انظر اليه وتعرف على اسم كل Pin فيه .

ملاحظه: لن نتعامل مع كل الـ Pin الموجوده بل سنستخدم فقط ثلاث منها سنفصلها لاحقاً

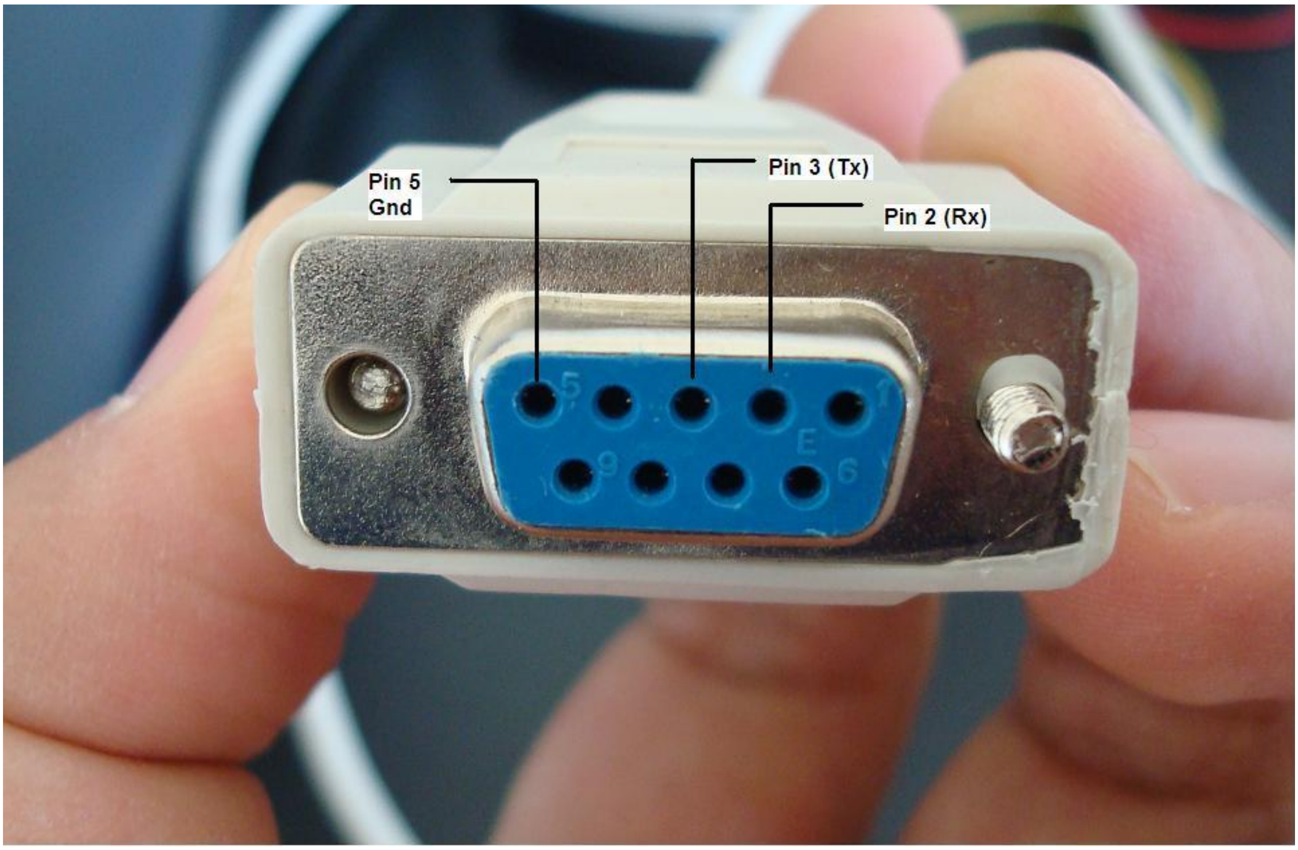


Pin	Signal	Pin	Signal
1	Data Carrier Detect	6	Data Set Ready
2	Received Data	7	Request to Send
3	Transmitted Data	8	Clear to Send
4	Data Terminal Ready	9	Ring Indicator
5	Signal Ground		

من الامور المهمه هنا هو معرفة طريقة الترقيم في هذا الكيبل وستلاحظ وجود ارقام دلالية على الترقيم على الكيبلات .... لان ببساطه اذا لم تعرف ذلك فسيكون ذلك من الصعب التعامل معه .

والان لنتعرف على الثلاث Pinالتي سنتعامل معهم .... لفهم ذلك لابد من التفكري في الآتي ...

نحن نريد التراسل بين المايكروكنترولر والكمبيوتر أي ان الطرف المرسل من المايكروكنترولر سيكون موصول مع الطرف المستقبل من الكمبيوتر والعكس صحيح ... وبهذا نحتاج الى طرفان هما Tx , Rx ونحتاج أيضا الى ارضي بينهما لذلك سنستخدم الـ GND وهذه الصورة تبين الاطراف التي سنتعامل معها بكل وضوح



اذا سنتعامل مع

Rx Pin2

Tx..................Pin3

GND.................Pin5

والان بعد ان تعرفنا على الـ serial Port سنتعرف على الـ Logic Level الخاص به .... فتحدثنا سابقاً بالنسبه للمايكروكنترولر يعتبر ان الـ صفر فولت هو logic low ويعتبر ان الـ 5V هي Logic High لكن في الـ Serial port الموضوع مختلف تماماً .... حيث يعتبر ان ما هو اقل من -3V وصولاً الى -25V هو Logic High ويعتبر ما هو اكبر من 3V وصولاً الى 25V هو Logic Low .... قد تضن انني قد أخطأت هنا ولكن كن واثقاً بما اقوله ..... هذا هو الـ Logic level الخاص بـ serial port هل ترى ما هو مدى الاختلاف اذاً بتأكيد نحن بحاجه الى دائره خارجية والا سيكون التفاهم معدوماً بين كل من الجهازين .

وبالمناسبه لابد من التطرق الى نوعين من الاتصال فهناك اتصال متزامن synchronous واتصال غير متزامن أي Asynchronous ولابد التفريق بينهما بشكل جيد .... لذلك فان الاتصال المتزامن باللغة البسيطه يمكن ان نقول بأنه كل فتره زمنيه محدده (تحدد لكل من المرسل والمستقبل) سيتم ارسال معلومه ما ... اذا الاتصال المتزامن يعتمد بشكل كبير على الزمن ..... اما الاتصال الغير متزامن فلا يعتمد على الزمن ..... لذلك في الاتصال الغير متزامن لابد من وجود دلاله تخبر المستقبل بأن الداتا ستصل بعد قليل ....

وبالمناسبه فالاتصال الذي سنتعامل معه اليوم هو الاتصال الغير متزامن .... لذلك لابد من وجود دلاله كما قلنا في الاعلى ... وهذه الدله بلغة علمية هي الـ Start Bit

وبلغة أبسط من ذلك .... في الاتصال المتزامن يكون هناك اتفاق بين المايكروكنترولر والكمبيوتر بحيث يقول المايكروكنترولر للكمبيوتر (انا سأبعث لك البيانات حرف حرف أو رقم رقم ... وسأنتظر مده ١٠ ملي ثانيه بين كل حرف) .... اما في الاتصال الغير متزامن فالاتفاق يكون غير ذلك سيكون بحيث يقول المايكروكنترولر للكمبيوتر (عندا اريد ان ارسل لك بيانات سأعطيك اشاره كذا مثلا) وهذه الاشاره هي الـ Start Bit

عذراً على هذه الاسلوب في الحديث لكن للأسف الشديد الكثير من الطلبه لا يستطيع التفرقه بينهما فهذه الطريقه الافضل لفهم الموضوع بكل وضوح .

لاحظ الشكل التالي يبين الـ serial data format ولاحظ وجود الـ start bit في البداية ومن ثم الـ Data وبعد ذلك هنا ايضاً Stop bit و أيضاً Parity bit الذي سنشرحه في الصفخ التالية



حيث يكون هذا الـ bit عند الكمبيوتر له قيمة واحد .... فعند ارسال البيانات يرسل المايكروكنترولر قبل البيانات قيمة صفر .... بالتالي تتغير الحاله لدى الكمبيوتر .... فيفهم الكمبيوتر ان هناك بيانات يجب استقبالها وستصل مباشره بعد الصفر .... فيبدأ الكمبيوتر بأخذ البيانات الى ٨ بت فقط .... فكيف يعرف الكمبيوتر ان البيانات قد انتهت .... لذلك لابد من وجود ما يسمى بـ Stop Bits .... ويتم كل ذلك خلال فتره زمنه قصير جداً.

اما بالنسبه للـ Parity bit .... فلا بد من التأكد من ان البيانات قد وصلت بشكل صحيح ... وهذا الـ Bit يستخدم لهذه الغايه .

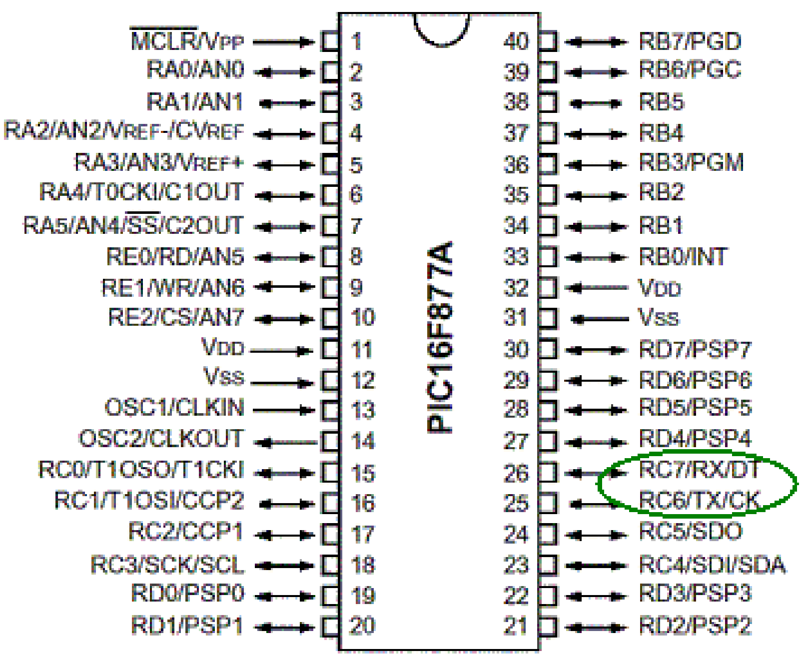
كل ما سبق كانت شروح نظرية عن الموضوع ... ما رأيكم بأن ندخل الى المتعه في عملية الاتصال بشكل عملي أكثر .... ولنبدأ الان

لابد لك عزيزي القارئ من التأكد من ان المايكروكنترولر الذي تملكه في فعلاً يمتلك خاصية الاتصال المتسلسل

وكيف ذلك؟

بسيط جداً انظر الى الـ 16f877a كما في الصورة .... لاحظ على الـ RC7 و RC6 ما المكتوب بجانبهما ؟ ستجد Rx و TX على التوالي

PDIP (40 pin)



مع العلم ان من خلال المايكروسي حتى ولو لم تكن المايكروكنترولر تحتوي على هذه الخاصيه يمكنك ان تصنعها بنفسك .... وهذا الموضوع .... قد أوعدكم ولكن قد نتحدث عنه في محاضرات قادمه

وفي لغة المايكروسي .... فالموضوع أبسط مما تتخيل ... فكل ما سبق قد لا تحتاجه ... ولكن لابد لك ان تفهمه جيدا .... لماذا؟؟ .... كي تكون مهندساً كهربائيه متمكناً مما من الموضوع التي تتحدث به .... فأنا أركز بشده ان يكون المتدرب قادراً حتى على اعطاء هذه الدوره بنفسه .

وبرنامج المايكروسي يختوي على مكتبه جاهزة تسمى بـ **USART Library** على دوال بسيطه تختصر هذا العمل بأكمله .... وهذه الدوال هي

```
UART1_Init(9600);
```

وهي مهمه لعملية التهيئه .... كما شرحنا ذلك في **ADC** فكل شيء له داله خاصه بتهيئه وهنا هذه هي الداله المستخدمه لتهيئة الـ **UART** .... لكن ما هو 9600؟

هذا عباره عن السرعه التي تنتقل بها البيانات ويسمى بـ **Baud Rate** ووحدته **bps** أي بت لكل ثانيه

والداله الاخره هي

```
Data= UART1_Read();
```

حيث ان كلمة **Data** هو متغير حرفي لابد من تعريفه مسبقاً في البرنامج ... وهذه الداله واضحه تعمل على قراءة البيانات الموجوده على الـ **Serial port** أي الموجوده على المداخل

وهي مشابها لدالة التي استخدمناها في الـ **ADC** التي كنا نقرأ القيمه الموجوده على مداخل الـ **ADC** راجع ذلك في مكانه

والداله الثالثة هي

```
UART1_wirte(Data);
```

وهذه الداله تستخدم لأرسال قيمه ما بحيث تكون القيمه أو الداتا بشكل عام مخزنه في المتغير **Data** ولو استخدمناها مع الجمله السابقه ..... سيريل نفس الداتا التي تم استقبالها لأنها تلقائياً خزنة في المتغير **Data**

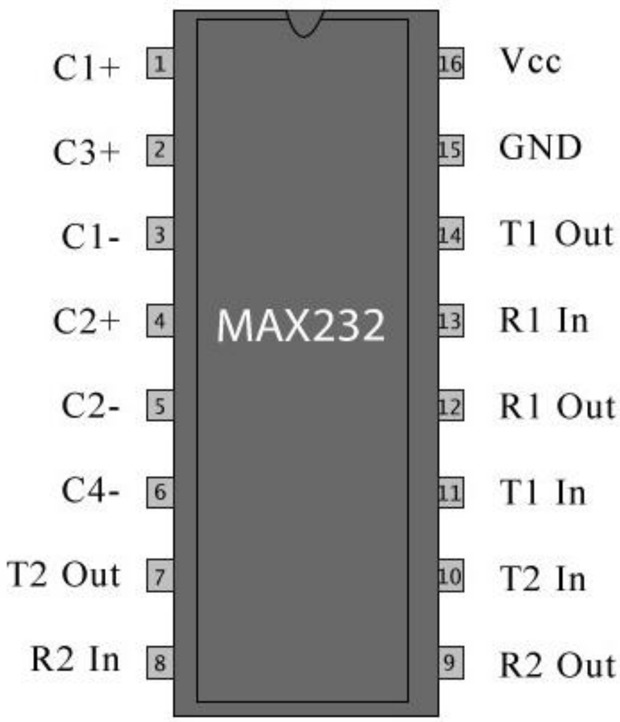
والداله الاخيره و المهمه التي لابد ان تفهمها جيدا هي موجوده على الصفحة التالي

```
If(UART1_data_ready()==1){
          هنا نكتب الاوامر التي يجب ان ينفذها المايكروكنترولر
          في حالة وصول الداتا له

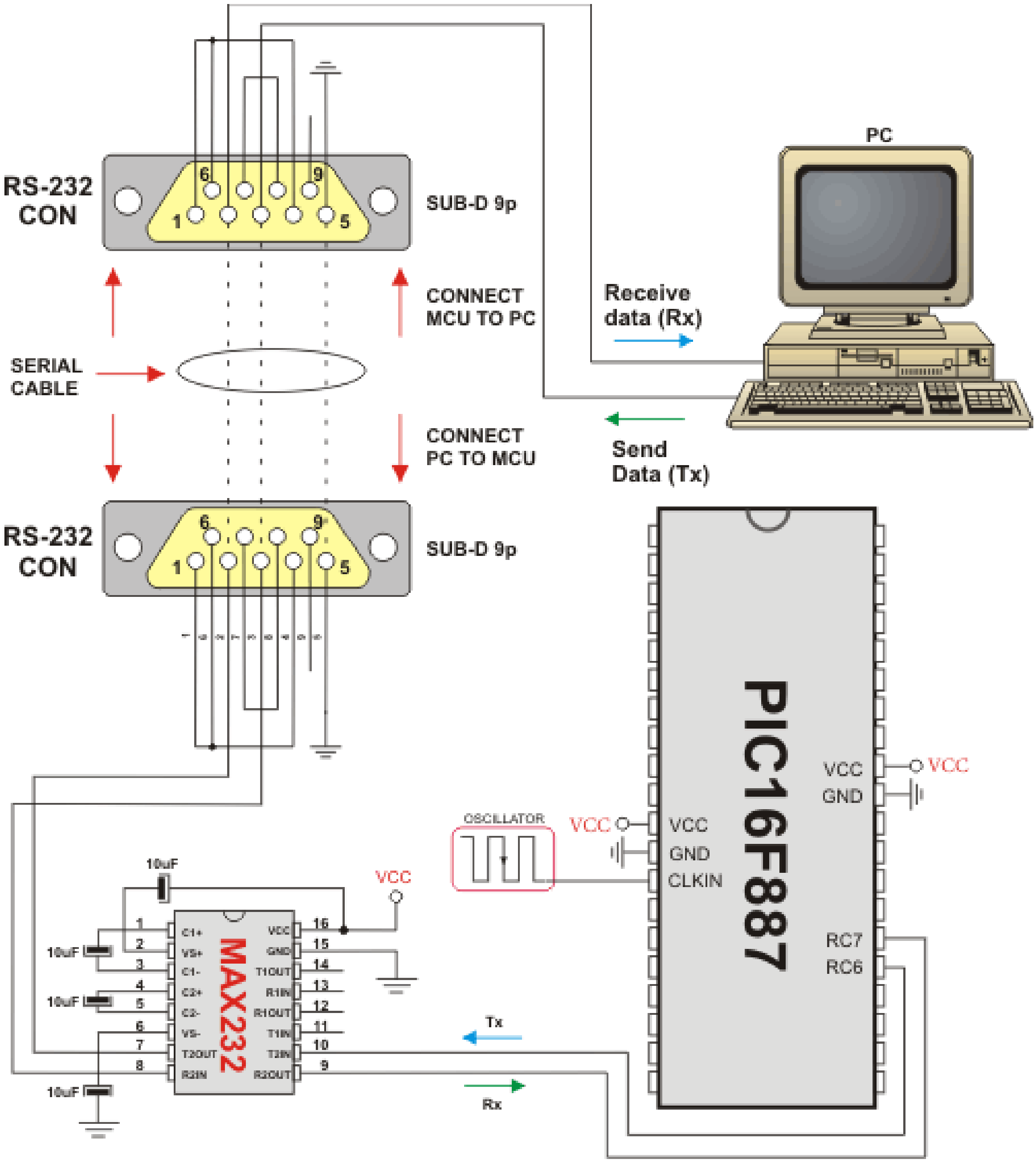
}
```

يكاد لا يخلو مشروع يتم استخدام به الاتصال المتسلسل من استخدام هذه الداله فهي تجعل البيك من ان يسأل نفسه ... هل وصلت الداتا على المداخل .... ولا بد من وضع هذه الجمله في احد الجمل التكراريه لتكرار السؤال على طول البرنامج .... أي كأنه يسأل عن وصول الـ Start Bit فاذا تأكد من وصول الداتا .... يعني هذا ان الشرط قد تحقق ... وبالتالي يقوم بتنفيذ الاوامر الموجوده بداخله ... كما ان نأمره بتخزينا ... أو التعديل عليها أو ان نضع جمله شرطيه اخرى بالداخل لتقارن الداتا اذا كانت مشابه لداتا اخره مخزنه به يتخذ القرار الفلاني وهكذا .....

قبل الانتقال الى عرض مثال على ذلك لابد من التعرف على الدائره المستخدمه لتحويل الـ Logic level وهي في غاية من الاهمية .... ونستخدم فيها مجموعه من المكثفات موصله بطريقة معينه مع قطعة الكترونيه اخرى متخصصه لهذه الغايه .... وهذه القطعة هي MAX232 وصوتها كما يلي

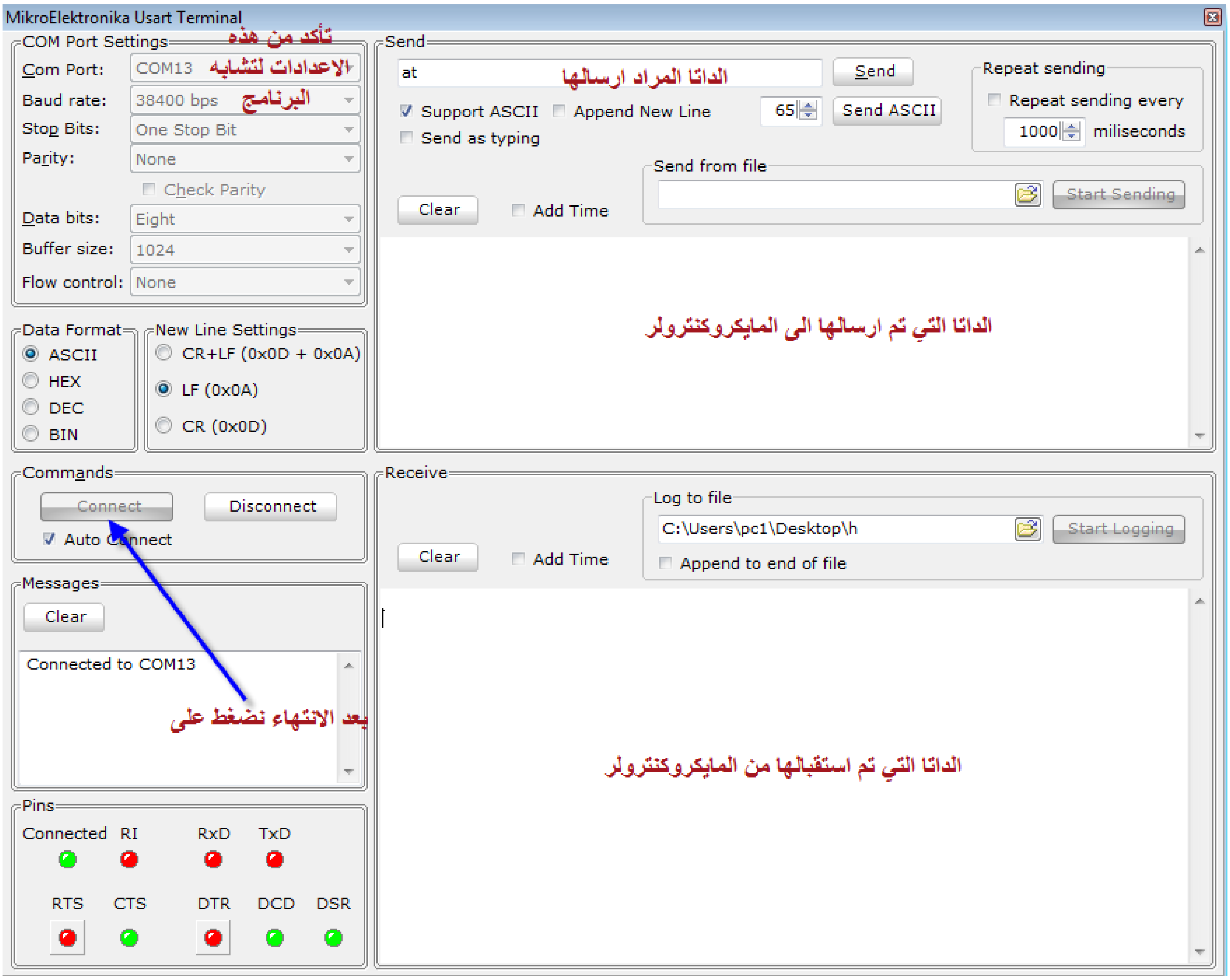


اما لطريقة التوصل هي كما في الدائرة التالية



والان هل فكرت ... كيف يمكن ان استقبل من المايكروكنترولر معلومه ما الى الكمبيوتر ؟
ما هو البرنامج الذي سأستخدمه في ذلك ؟؟

في الواقع هناك الكثير من البرمج وحدها برنامج موجود على المايكروسي نحصل عليه من القائمة Tool ونختار الـ USAR Terminal .... فيضهر لنا البرنامج التالي

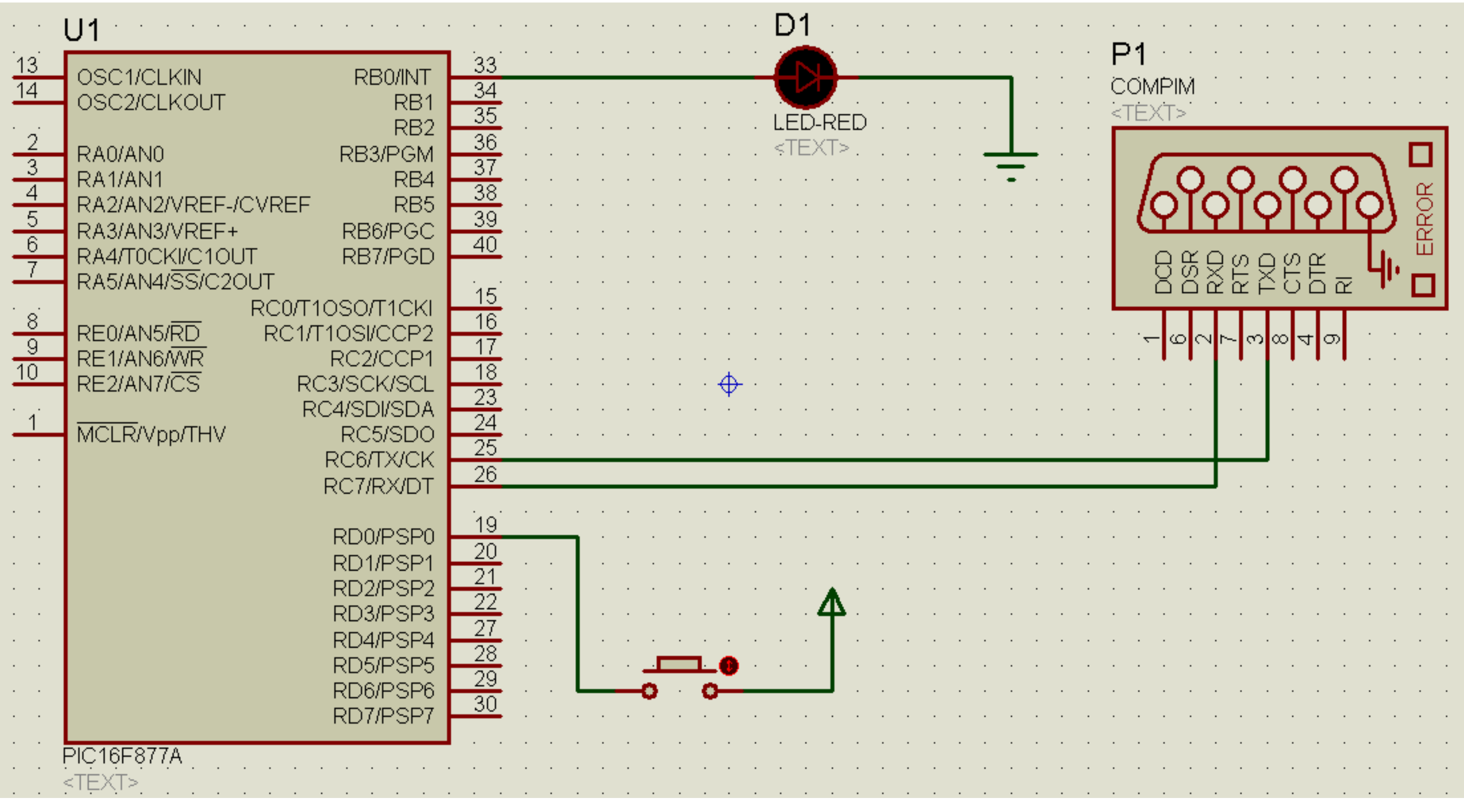


يمكن من خلال هذه الشاشه ... ان تتم عملية المراسله في البيانات بين المايكروكنترولر والكمبيوتر ... ولكن للأسف الشديد يجب ان يكون المايكروكنترولر موجود فعلياً وموصل بالدائره السابقه ... وتوصيل الكيبل بالكمبيوتر ثم النقر على الزر Connect بعد التأكد من ان جميع الاعدادات السابقة هي مطابقه تمام لما هو في البرنامج مثل الـ baud rate ورقم الـ com الموصل عليه الكيبل بالكمبيوتر

لكن الا يوجد طريقه لعمل محاكاه لدائره قبل قبل تنفيذها بشكل حقيقي ؟؟؟...... بطبع يوجد

فيمكن ان نعمل اتصال بين برنامج البروتس والكمبيوتر ... من خلال استخدام برنامج لصنع serial port افتراضي ... وهنا سنتعامل مع البرنامج VSPE وسنشرح التعامل معه من خلال المثال التالي

قم برسم الدائرة البسيطه التالية على برنامج بروتس



لاحظ اننا لم نستخدم الدائرة التي تحدثنا عنها سابقاً لان البرنامج هدفه هو عمل محاكاه فاذا رسمنا الدائره أو لم نرسمها ستكون نفس النتيجه .... لذلك لاحظ اننا وصلنا المرسل للمايكروكنترولر مع المستقبل لـ **serial port** والعكس صحيح أيضاً .

حاول ان تستخدم الدائرة السابقة وشاهد النتيجه ... التي يجب ان تتطابق مع النتيجة التالية

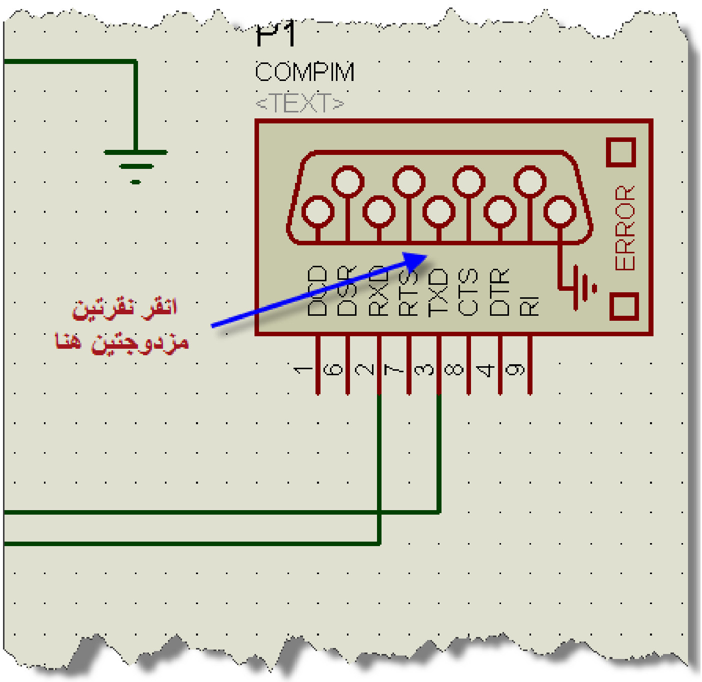
والان ننتقل الى الكود الذي سنبرمج المايكروكنترولر به كما في الصفحة التالية

الكود:

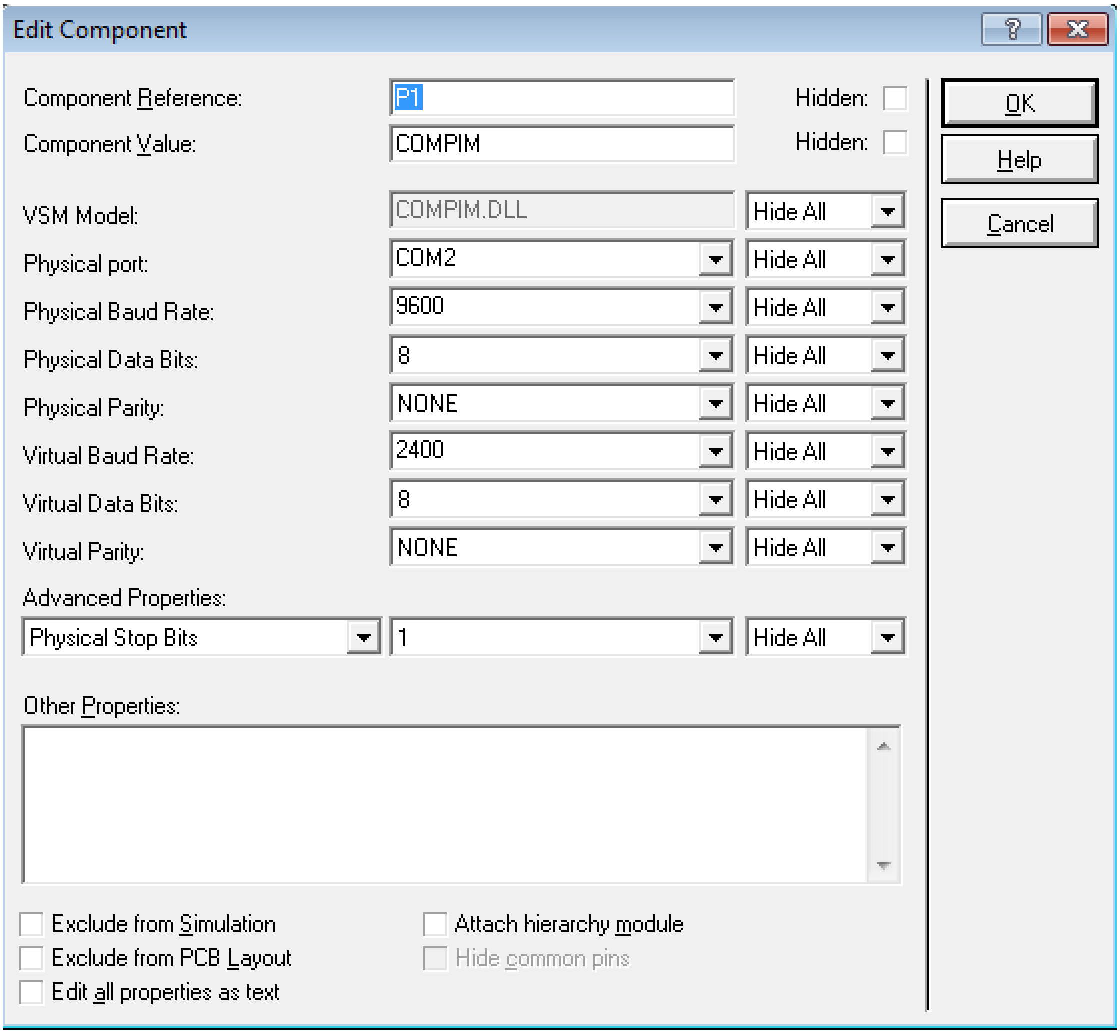
```
unsigned short i;
void main()
{
Uart1_Init(2400);
trisb=0;
portb=0x00;
portd=0xff;
loop:
{
if(portd=0x01)
{ Uart1_write('3');
delay_ms(500);}
 if (Uart1_Data_Ready())
 {i = Uart1_Read();
if(i==97){portb=0xff;}
if(i==50){portb=0x00;}
}goto loop
}
}
```

والان ننتقل بعد القيام بعملية ضبط قيمة المتذبذب ... وادخال البرنامج ... سنستخدم البرنامج VSPE للقيام بعمل Serial port وهمي ... لذلك اتبع الخطوات التالية كما يلي

انقر نقرتين مزدوجتين على المكان المؤشر له بالسهم



الان ستضهر لك القائمه التالية اضبط الاعدادات كما في الشكل التالي

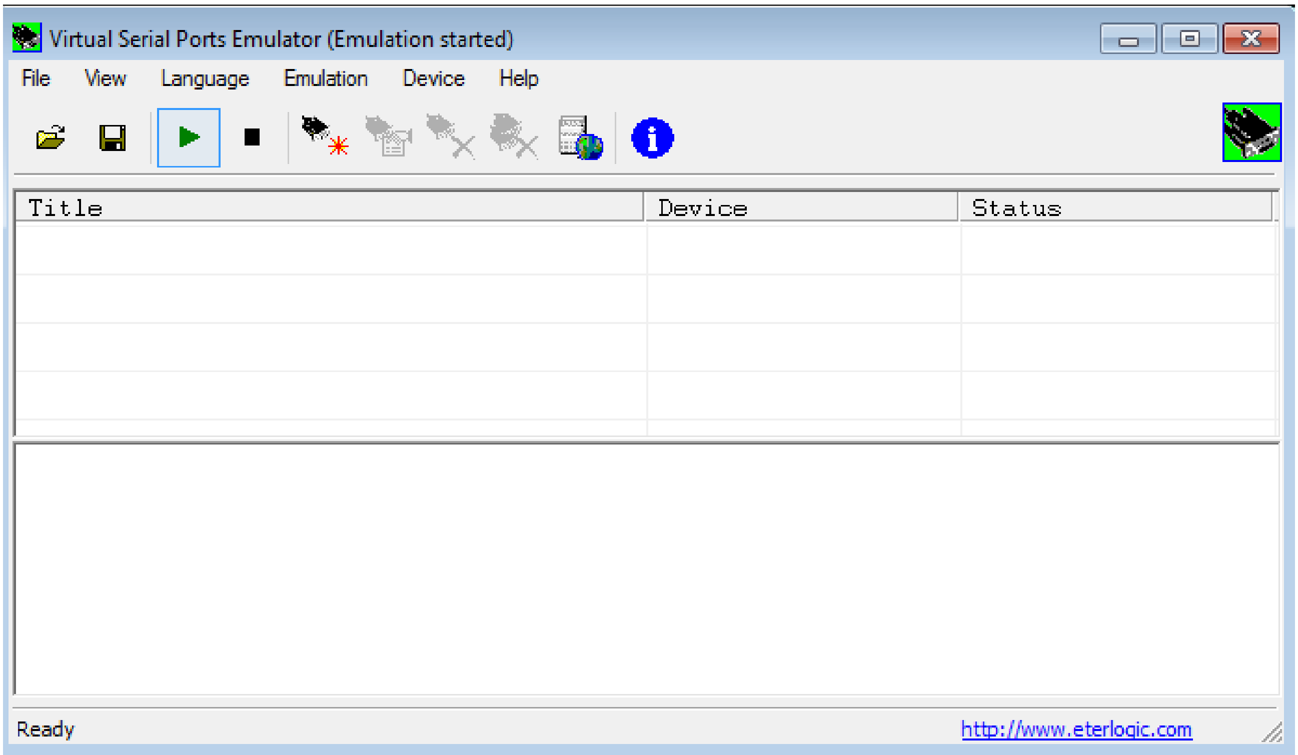


وبعد الانتهاء انقر على OK .... وبعد ذلك قم بتشغيل البرنامج VSPE  **لتظهر الشاشه الرئيسيه له كما في الصفحة التالية ....**

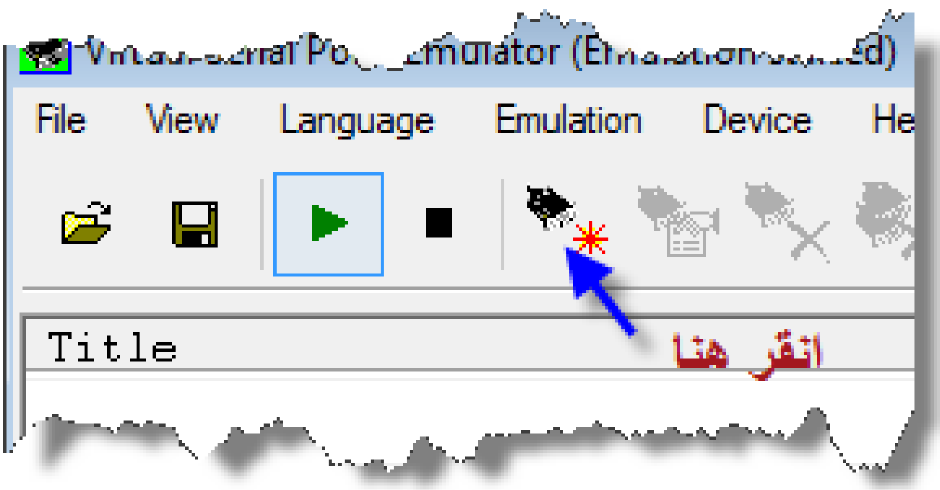
وكما ذكرنا هذا البرنامج سنستخدمه فقط لعمل المحاكاه .... أي بدل من توصيل الكيبل السيريال بشكل حقيقي يمكن ان نستعمل هذا البرنامج لعمل اتصال تخيلي

مع العلم ان البرنامج موجود مع ملف المحاضره هو وملف المحاكاه وحتى البرنامج .... ليسهل عليك فهم الموضوع

هذه هي الشاشة الرئيسيه للبرنامج VSPE

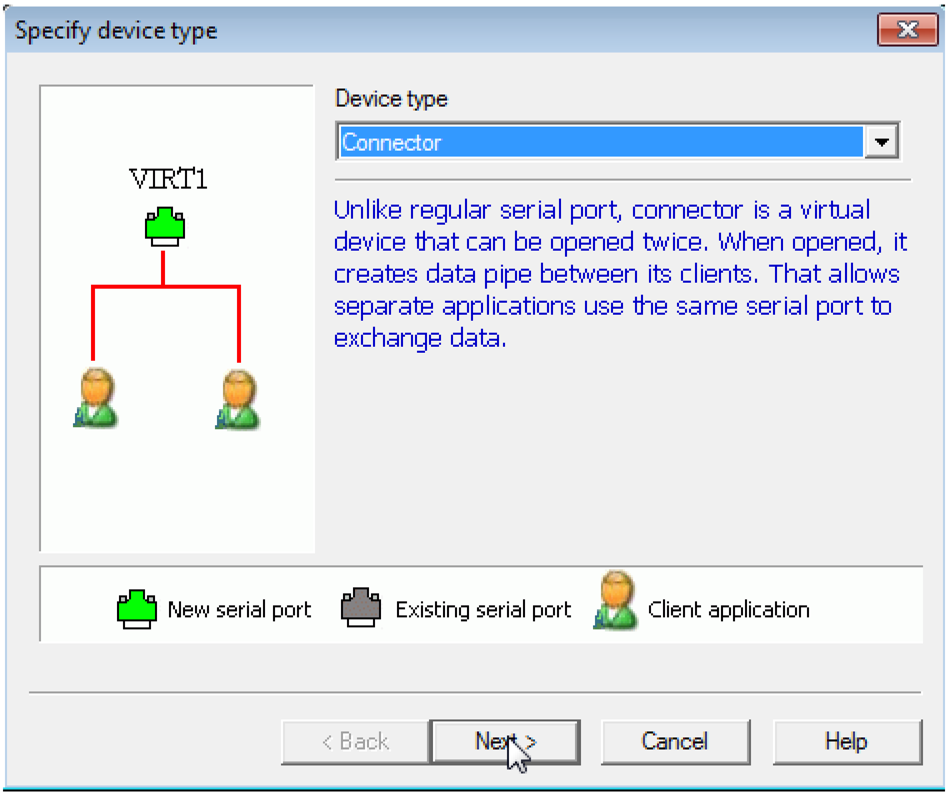


والان انقر على هذه الايقونه في البرنامج

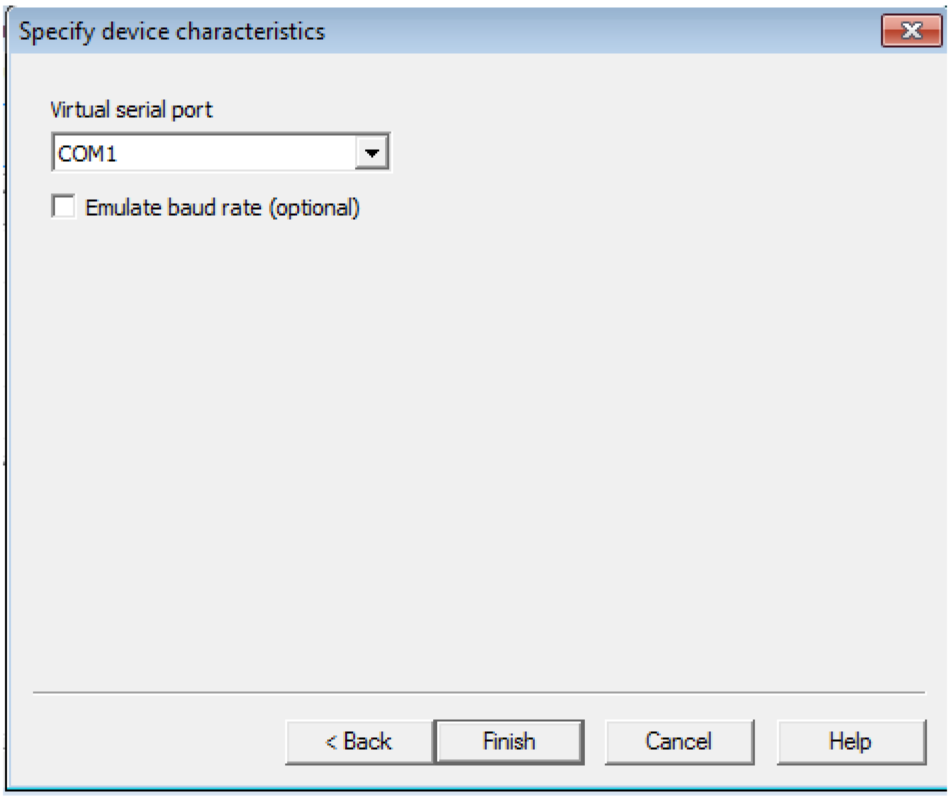


عندها ستظهر لك الشاشة التالية

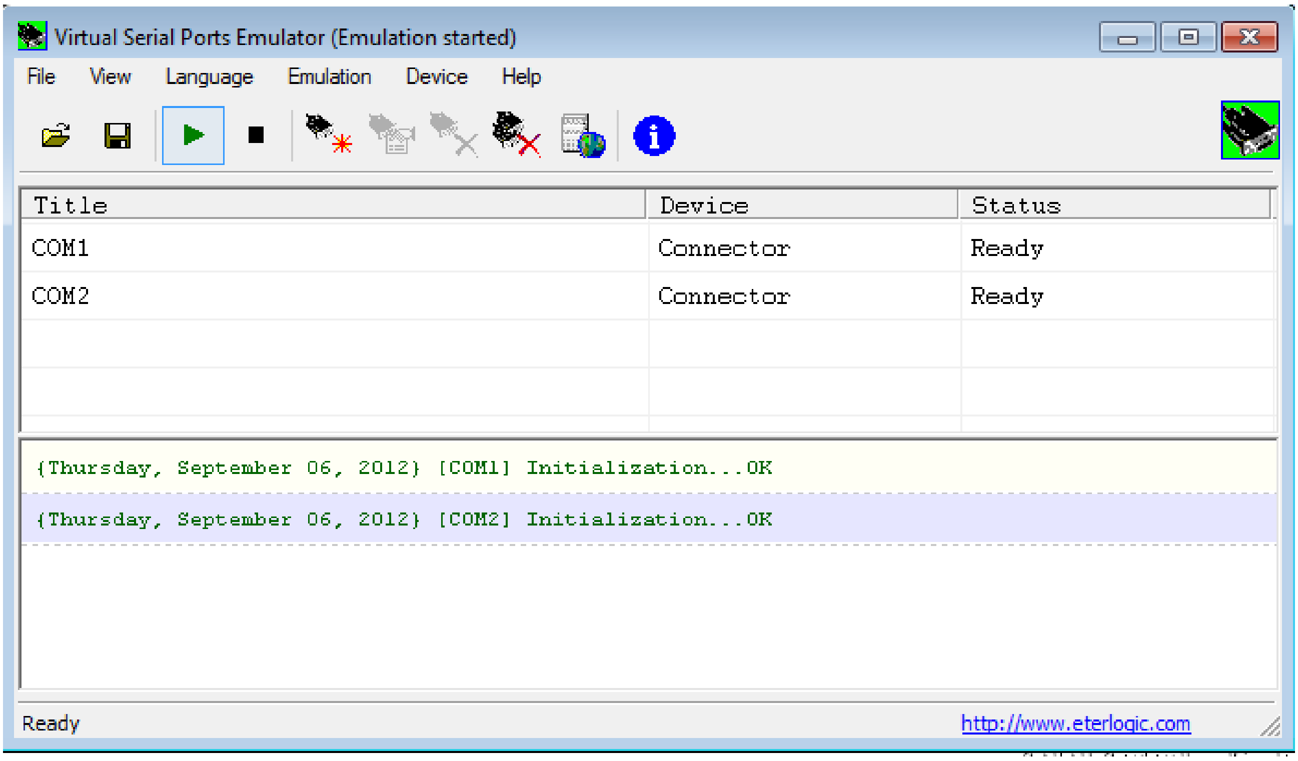
انقر على الزر next مباشره



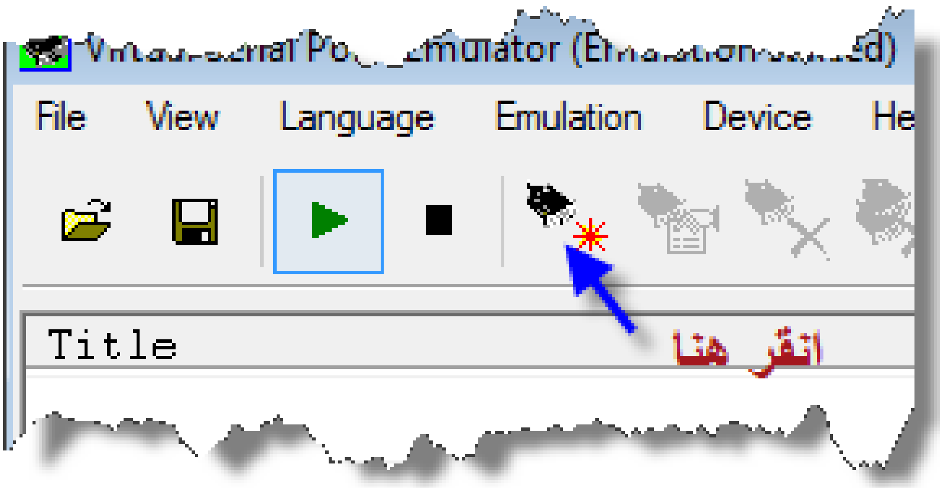
ستظهر الشاشه التالية اختر منها Com1 وانقر على Finish



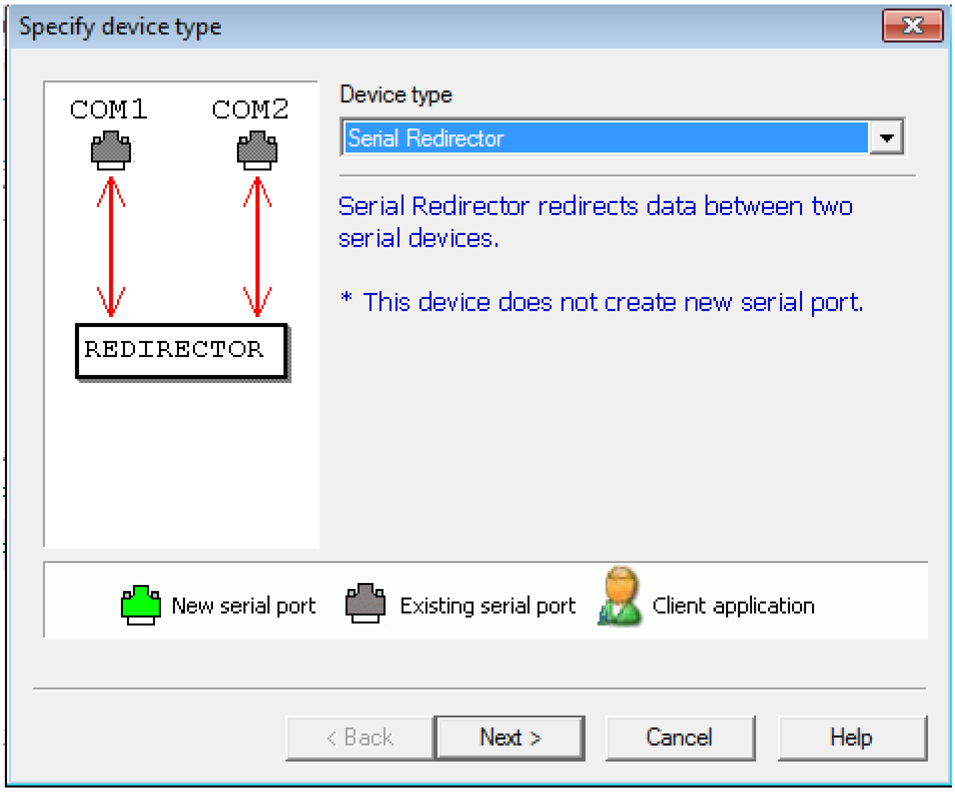
ثم اعد تكرار نفس العمليه لكن عند الوصول الى الخطوة الاخير اختر com2 وثم finish لنحصل على الشاشة التالية



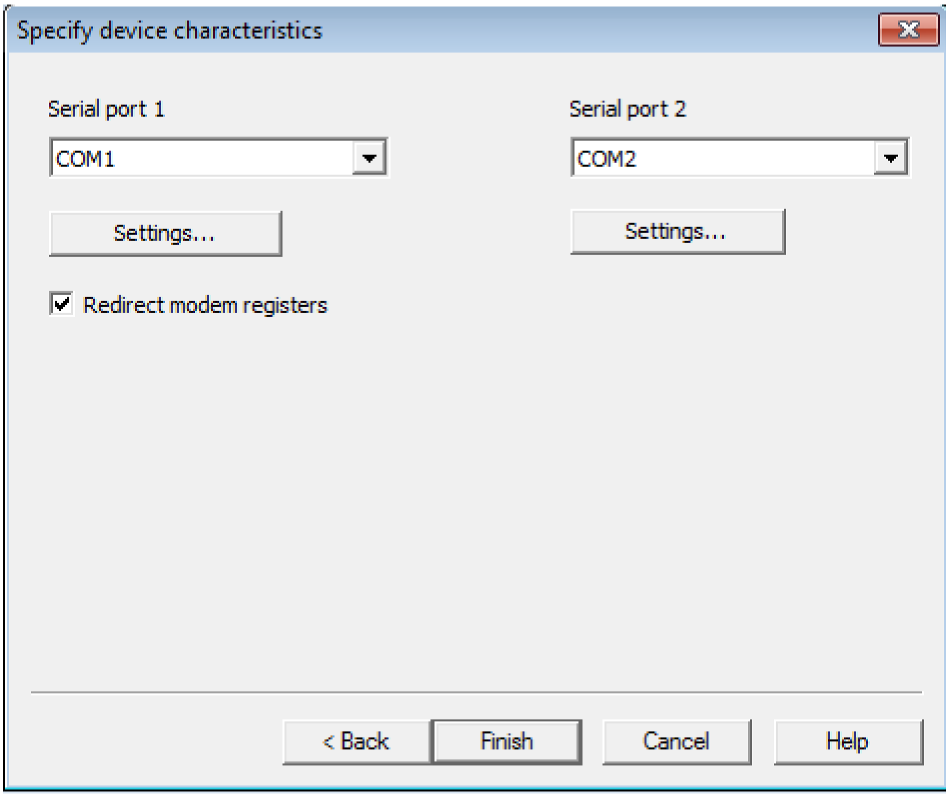
والان انقر مره اخرى على



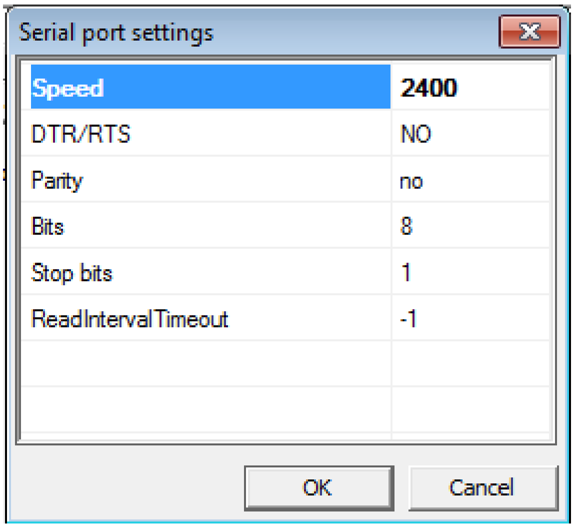
واختر الخيار serial redirector وثم انقر على next



لتظهر هذه الشاشه التالية اختر منها settings



بعد النقر على settings اضبط الخيارات كما في الصورة التالية



ومن ثم انقر على **finish** وبهذا نكون قد انهينا الاتصال فنعود الان لنستخدم برنامج الـ **Usart terminal**

قد يبدو الامر معقداً في البداية .... لكن كرر قراءتك للملف مره ومرتان وثلاث مرات
سترى ان ذلك في غاية من الروعة

ولتسهيل ذلك عليك قمت بعمل هذا الفيديو الذي يوضح عملية المحاكاه لدائرة بشكل مبسط
تابع الفيديو ...... من المرفقات في المحاضرة

انتهت المحاضرة التاسعة